

***Rotman***

# A QUICK INTRO TO R

R Workshop (M/E MBA)

July 20, 2021 Prepared by Jay Cao / [TDMDAL](#)

Website: <https://tdmdal.github.io/r-workshop-2021-memba/>



Rotman School of Management  
UNIVERSITY OF TORONTO

# Goal for Today – Answer Three Questions

- What's R?
- What can I use R for?
- How to learn R (on my own)?

# Plan for Today

- Intro to Intro
  - What is R and what can R do?
  - Setup R
  - Motivation examples
- How to learn R and a quick walk-through
  - Basics of R programming
  - Data science with R
- Learning Road Map and Resources

# What's R?



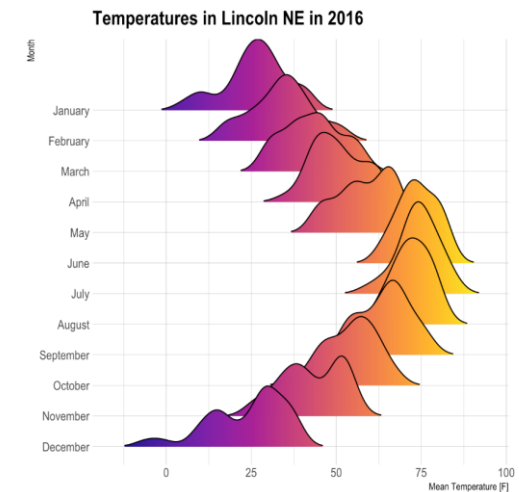
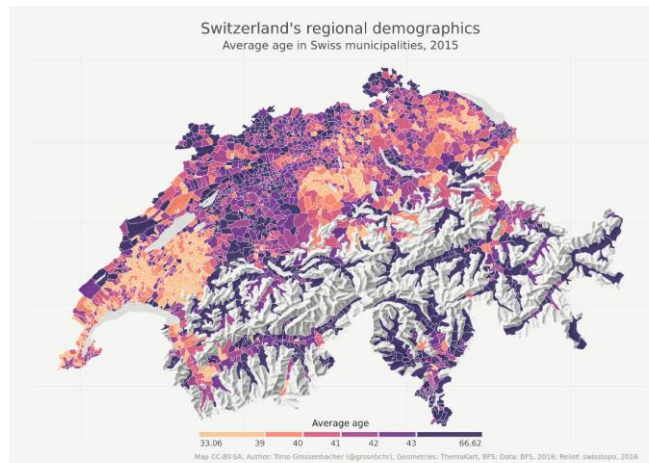
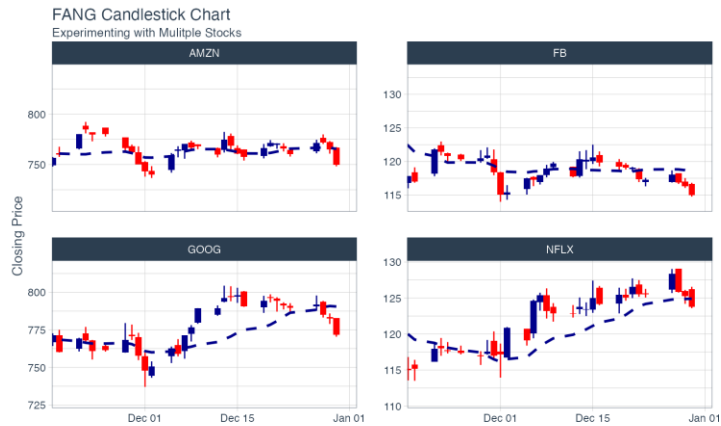
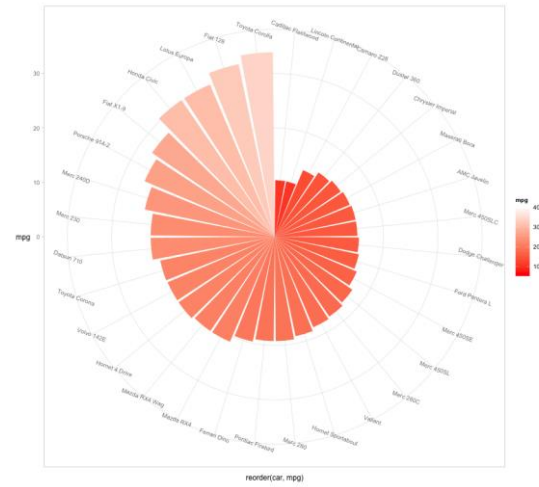
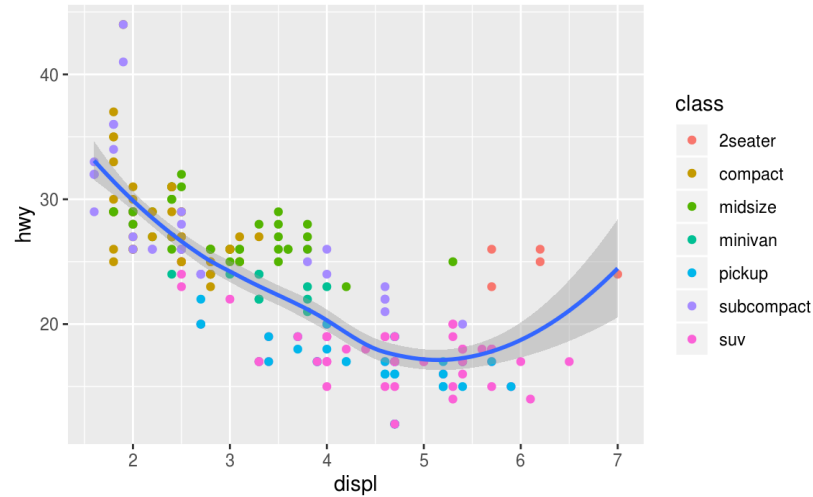
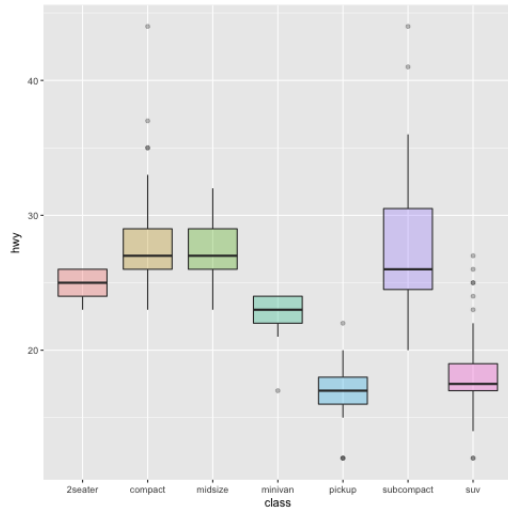
- R = a language + an eco-system
  - A free and open-source programming language
  - An eco-system of many high-quality user-contributed libraries/packages
- In the past R is mostly known for its statistical analysis toolkits
- Nowadays R is capable of (and very good at) many other tasks
  - Tools that cover the whole data analysis workflow
  - Tools for web technology...

# What can R do – Statistics & related

- Statistics & Econometrics
  - Regressions
  - Time series analysis
  - Bayesian inference
  - Survival analysis
  - ...
- Numerical Mathematics
  - Optimization
  - Solver
  - Differential equations
  - ...
- Finance
  - Portfolio management
  - Risk management
  - Option pricing
  - ...
- ...

See more R Empirical Finance Packages on [R Task View - Finance](#)

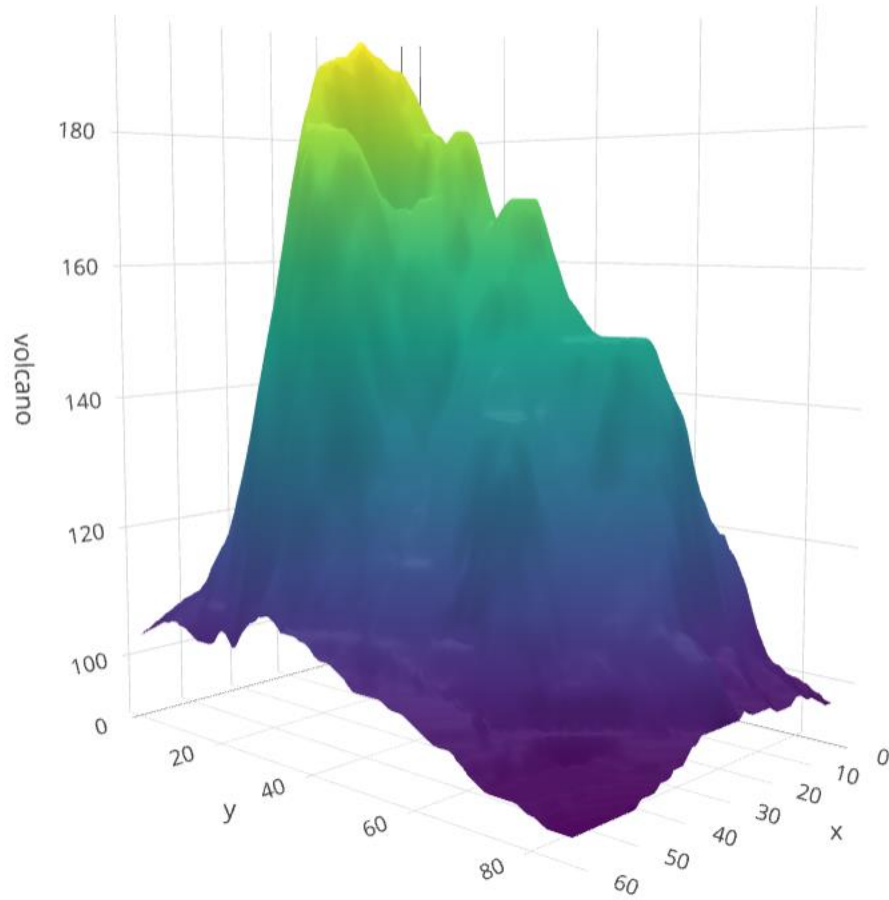
# What can R do – Graphics (static ones)



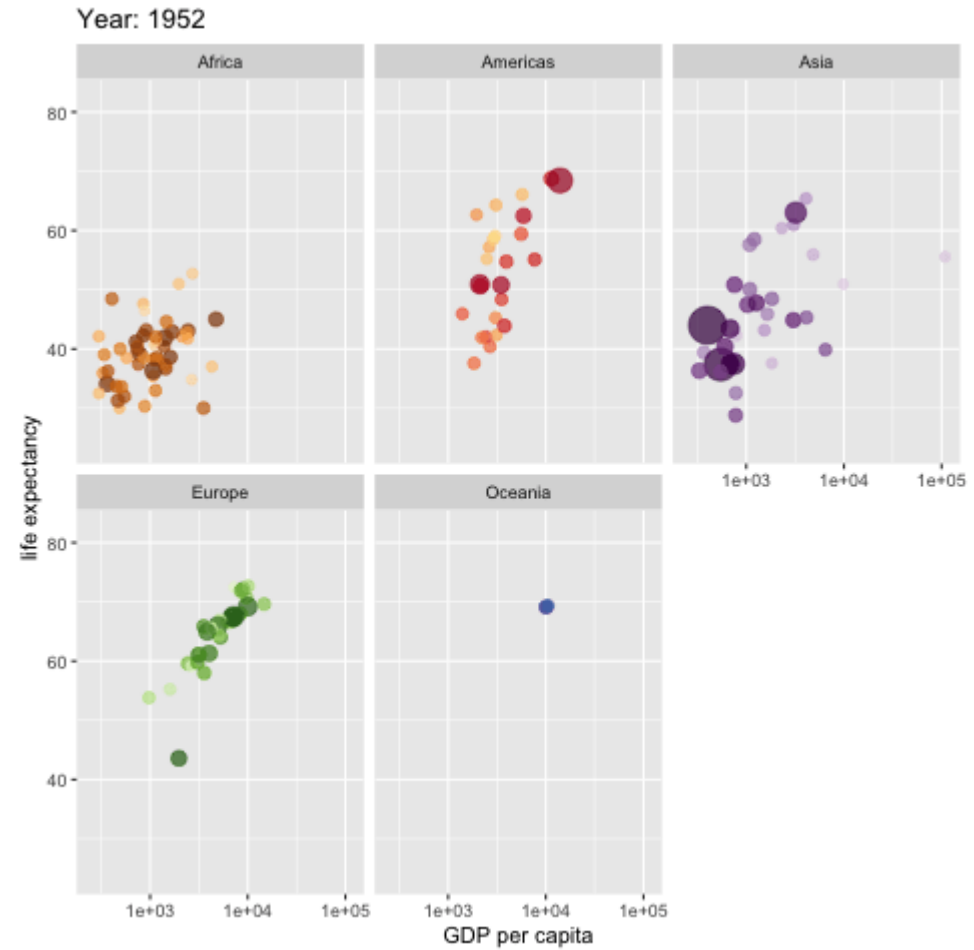
<https://www.r-graph-gallery.com/>

[https://timogrossenbacher.ch/2016/12/beautiful-thematic-maps-with-ggplot2-only/;](https://timogrossenbacher.ch/2016/12/beautiful-thematic-maps-with-ggplot2-only/)

# What can R do – Graphics (dynamic ones)



[https://plot.ly/r/3d-surface-plots/;](https://plot.ly/r/3d-surface-plots/)

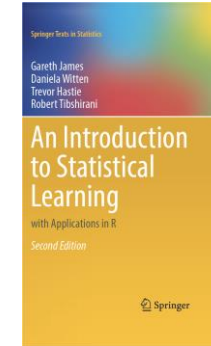


<https://github.com/thomasp85/gganimate;>

# What can R do – Others (1)

- Machine learning

- Statistical learning (clustering, decision tree, etc.)
  - [An Introduction to Statistical Learning \(with Applications in R\)](#)



- Deep learning (neural networks)

- Interface to [Keras](#) and [Tensorflow](#) (via [reticulate](#), an R to Python interface)
- [Torch for R](#) (natively from R; similar as PyTorch in Python)



- Natural language processing (e.g., [tidytext](#), [topicmodels](#))

1. See more R Machine Learning Packages on [R Task View - ML & Statistical Learning](#)
2. See more R Natural Language Processing Packages on [R Task View - NLP](#)



# What can R do – Others (2)

- Web technology
  - Web scraping (e.g. [rvest](#))
  - API wrapper (e.g. Twitter: [rtweet](#); bigquery: [bigrquery](#); Quandl: [Quandl](#))
  - Shiny web app (<https://shiny.rstudio.com/>)
- Reporting
  - [R Markdown](#) (write reports, slides, blogs, books, etc. See a gallery [here](#).)
- ... (see [R Task View](#) for more)

# R vs Excel and BI Tools vs Python



- Excel & Business Intelligence (BI) Tools (e.g., Tableau, Power BI, etc.)

- 2-D tables as basic data structure
- Good UI (User Interface) and minimum programming
- Limited modeling tools
- Not easy to reproduce an analysis (because it's hard to store UI clicks)
- Not flexible enough for complicated analysis problems, i.e., problems with
  - Many data cleaning steps/pipelines
  - Many different models to try



Power BI Desktop

- Python

- Very similar to R, but R is more specialized in data analysis
- R is much easier to learn (in my opinion)



# Why learn R (What can R do for You)?

- Beyond Excel Data Analysis
  - I wish Excel could...
- Automate boring repeating tasks
  - e.g., daily data collection from different sources, weekly dashboard update
- Prototype ideas
  - e.g., a novel trading strategy, a new credit risk model
- Really, find anything that interests you and use R...

# Plan for Today

- **Intro to Intro**
  - What is R and what can R do?
  - **Setup R**
  - Motivation examples
- How to learn R and a quick walk-through
  - Basics of R programming
  - Data science with R
- Learning Road Map and Resources

# Setup R

- R & RStudio on your computer (most of you should choose this one)
  - Install R (<https://www.r-project.org/>)
  - Install RStudio (<https://rstudio.com/products/rstudio/download/>)
- R & RStudio in the Cloud (run R without installation)
  - RStudio Cloud (<https://rstudio.cloud/>)
  - UofT JupyterHub RStudio (<https://jupyter.utoronto.ca/hub/login>)
- R & **Notebook** in the Cloud (run R without installation)
  - UofT JupyterHub Notebook (<https://jupyter.utoronto.ca/hub/login>)
  - Google Colab (<https://colab.to/r>)

# What's RStudio?



The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for creating a graph. The code uses `tribble` to define edges, `distinct` and `rename` to process nodes, and `create_graph` to build the graph object `g`.
- Environment Pane:** Shows the 'Global Environment' with a 'Data' section containing:

Object	Description
edge_tb	3 obs. of 2 variables
g	List of 12
node_tb	4 obs. of 1 variable
node_tb_tp	2 obs. of 1 variable
raw	4 obs. of 5 variables
- Viewer Pane:** Displays a directed graph with four nodes (1, 2, 3, 4) and three edges: 1 → 2, 2 → 3, and 2 → 4.
- Console:** Shows the execution of the code, including the `render_graph()` command.

# RStudio Cloud

The screenshot displays the RStudio Cloud web interface. The browser address bar shows the URL `https://rstudio.cloud/spaces/112457/project/2046604`. The interface includes a sidebar on the left with navigation options like 'Spaces', 'Your Workspace', 'R Intro', 'New Space', 'Learn', 'Guide', 'What's New', 'Primers', 'Cheat Sheets', 'Help', 'Current System Status', 'RStudio Community', and 'Info'. The main workspace is titled 'R Intro / Workshop 1' and features a menu bar with 'File', 'Edit', 'Code', 'View', 'Plots', 'Session', 'Build', 'Debug', 'Profile', 'Tools', and 'Help'. Below the menu is a toolbar with icons for file operations and a 'Go to file/function' search bar. The central editor shows an 'Untitled1' script with a single line of code: `1 |`. The bottom panel contains a 'Console' with the following text: 

```
/cloud/project/  
R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.  
  
R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.  
  
Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.  
  
> |
```

 The right-hand side of the interface has three panels: 'Environment' (showing 'Global Environment' and 'Environment is empty'), 'Files' (showing a file browser with a table of files), and 'Plots' (empty). The 'Files' panel table is as follows:

	Name	Size	Modified
	..		
<input type="checkbox"/>	.Rhistory	0 B	Dec 28, 2020, 4:52 PM
<input type="checkbox"/>	project.Rproj	205 B	Dec 28, 2020, 4:52 PM


# RStudio at UofT Jupyterhub

 Jupyter Notebook  RStudio  JupyterLab'. There is an orange 'Log in to start' button and a link to 'JupyterHelp' for support. At the bottom, there is a welcome message and logos for Jupyter and RStudio."/>

JupyterHub

https://jupyter.utoronto.ca/hub/login

Not syncing



UNIVERSITY OF  
**TORONTO**  
JUPYTERHUB

Operated by 2i2c.org



After logging in, open:  Jupyter Notebook  RStudio  JupyterLab

Log in to start

Use [JupyterHelp](#) to open tickets for support questions.

Welcome to the new University of Toronto **JupyterHub for Teaching** site.

A proof of concept service, developed as a partnership between the [Office of the CIO](#) (Information Technology Services), the Faculty of Arts & Science's new Computational and Data Science





# R Notebook in Google Colab



The screenshot shows a Google Colab notebook interface. The browser address bar displays the URL: `https://colab.research.google.com/github/tdmdal/r-workshop-researchers/blob/master/docs/rn1_A_Simple_Regression.ipynb`. The notebook title is "rn1 A Simple Regression". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with options like "+ Code", "+ Text", and "Copy to Drive", and a status bar showing RAM and Disk usage. The notebook content is as follows:

## 1. Data Import and Manipulation

We first import a dataset from the workshop website. This is a dataset on married women labor force participation used in [Mroz 1987](#). The dataset is also used throughout Wooldridge's text book: Introductory Econometrics: A Modern Approach. After briefly inspecting the data, we create a new column `lwage` in preparation for a simple regression.

```
[ ] # load data
data_url <- "https://tdmdal.github.io/r-workshop-researchers/data/mroz_1987.csv"
mroz_1987 <- read.csv(data_url)
```

[ ] # take a look at the structure of the data  
`str(mroz_1987)`

See a description of the data columns [here](#).

```
[ ] # print the first few rows of the dataset
head(mroz_1987)
```

```
[ ] # create log wage
mroz_1987["lwage"] <- log(mroz_1987["wage"])
```

## 2. Modelling

We will run a simple regression to investigate return on education for married women:  $\log(wage) = \beta_0 + \beta_1 educ + u$ .


```
[ ] # setup a regression model
lr <- lm(formula = lwage ~ educ, data = mroz_1987)
```

# R Notebook at UofT Jupyterhub

JupyterHub

https://jupyter.utoronto.ca/hub/login

Not syncing



UNIVERSITY OF  
**TORONTO**  
JUPYTERHUB

Operated by [zi2c.org](https://zi2c.org)



After logging in, open:  Jupyter Notebook  RStudio  JupyterLab

Log in to start

Use [JupyterHelp](#) to open tickets for support questions.

Welcome to the new University of Toronto **JupyterHub for Teaching** site.

A proof of concept service, developed as a partnership between the [Office of the CIO](#) (Information Technology Services), the Faculty of Arts & Science's new Computational and Data Science



# Plan for Today

- **Intro to Intro**
  - What is R and what can R do?
  - Setup R
  - **Motivation examples**
- How to learn R and a quick walk-through
  - Basics of R programming
  - Data science with R
- Learning Road Map and Resources

# A Few Examples

- Analyze portfolio performance
- Look for trends in R community through Twitter
- Recognize handwritten digits - an example of deep learning



**PerformanceAnalytics  
Package**



**K Keras**

**TensorFlow**

# A Few Examples: What to Look For

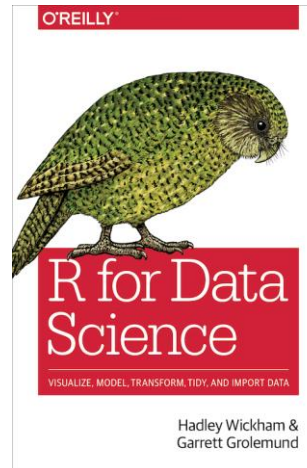
- Focus on analysis workflow (by reading the code comments)
  - Import and manipulate data
  - Model data
  - Report and visualize results
- Don't focus on R syntax
- Do notice everything is done in a sequential way
  - no conditional branching or looping

# Plan for Today

- Intro to Intro
- **How to learn R** and a quick walk-through
  - Basics of R programming
  - Data science with R
- Learning Road Map and Resources

# How to Learn R

- Step 1. Basics of R programming
  - Data and programming structures (learn R syntax; relatively easy)
  - How to turn ideas into code (**hard; takes time and practice**)
- Step 2. Data Science with R
  - Data wrangling
  - Modeling
  - Reporting and visualization
- A Good Learning Approach
  - Learn underlying principles (e.g., why organize data in a certain way)
  - Learn best practices (e.g., follow a consistent analysis workflow)



Free books: [Hands-On Programming with R](#); [R for Data Science](#)

# Plan for Today

- Intro to Intro
- How to learn R and a quick walk-through
  - Basics of R programming
    - Expression & Assignment
    - Data Structure
    - Programming Structure (control flow & function)
    - Turn ideas into code
  - Data science with R
- Learning Road Map and Resources



# Expression and Assignment

```
# expression
```

```
2 + sqrt(4) + log(exp(2)) + 2^2
```

```
# assignment
```

```
x <- 3
```

```
y <- (pi == 3.14)
```

# R Data Structure - Overview

	Homogeneous	Heterogeneous
1-d	<b>Atomic vector</b>	<b>List</b>
2-d	Matrix	<b>Data frame</b>
n-d	Array	

# R Data Structure - Overview

	Homogeneous	Heterogeneous
1-d	<b>Atomic vector</b> →	<b>List</b>
2-d	Matrix	↓ <b>Data frame</b>
n-d	Array	

# Atomic Vectors

```
# create R vectors
```

```
vec_character <- c("Hello,", "World!")
```

<b>Hello,</b>	<b>World!</b>
---------------	---------------

```
vec_integer <- c(1L, 2L, 3L)
```

<b>1</b>	<b>2</b>	<b>3</b>
----------	----------	----------

```
vec_double <- c(1.1, 2.2, 3.3)
```

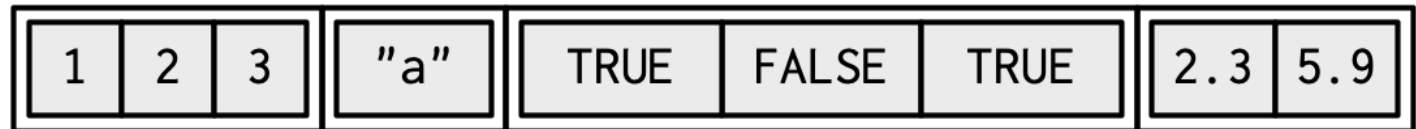
<b>1.1</b>	<b>2.2</b>	<b>3.3</b>
------------	------------	------------

```
vec_logical <- c(TRUE, TRUE, FALSE)
```

<b>TRUE</b>	<b>TRUE</b>	<b>FALSE</b>
-------------	-------------	--------------

# List

```
# create an R list
l1 <- list(
  1:3,
  "a",
  c(TRUE, FALSE, TRUE),
  c(2.3, 5.9)
)
```



# Data Frame

```
# create a data frame
df1 <- data.frame(
  x = 1:3,
  y = letters[1:3],
  z = c(1.1, 2.2, 3.3)
)
```

<b>x</b>	<b>y</b>	<b>z</b>
1	"a"	1.1
2	"b"	2.2
3	"c"	3.3

# Data Frame

```
# create a data frame
df1 <- data.frame(
  x = 1:3,
  y = letters[1:3],
  z = c(1.1, 2.2, 3.3)
)
```

x	y	z
1	"a"	1.1
2	"b"	2.2
3	"c"	3.3

# Data Frame

```
# create a data frame
df1 <- data.frame(
  x = 1:3,
  y = letters[1:3],
  z = c(1.1, 2.2, 3.3)
)
```

x	y	z
1	"a"	1.1
2	"b"	2.2
3	"c"	3.3



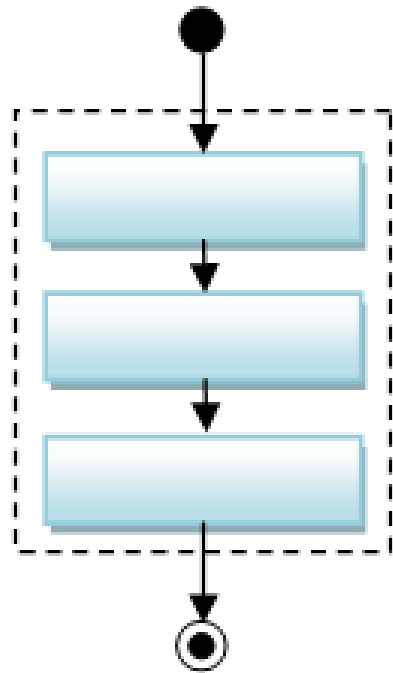
# A Cousin to Data Frame - Tibble

```
# load tibble library (part of tidyverse lib)
library(tibble)

# create a tibble
tb1 <- tibble(
  x = 1:3,
  y = letters[1:3],
  z = c(1.1, 2.2, 3.3)
)
```

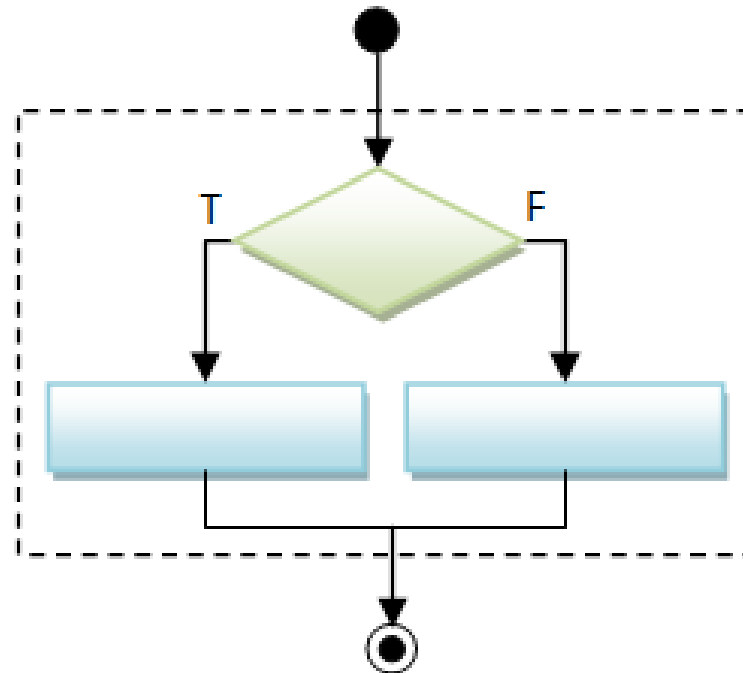
x	y	z
1	"a"	1.1
2	"b"	2.2
3	"c"	3.3

# Programming Structure: Control Flows



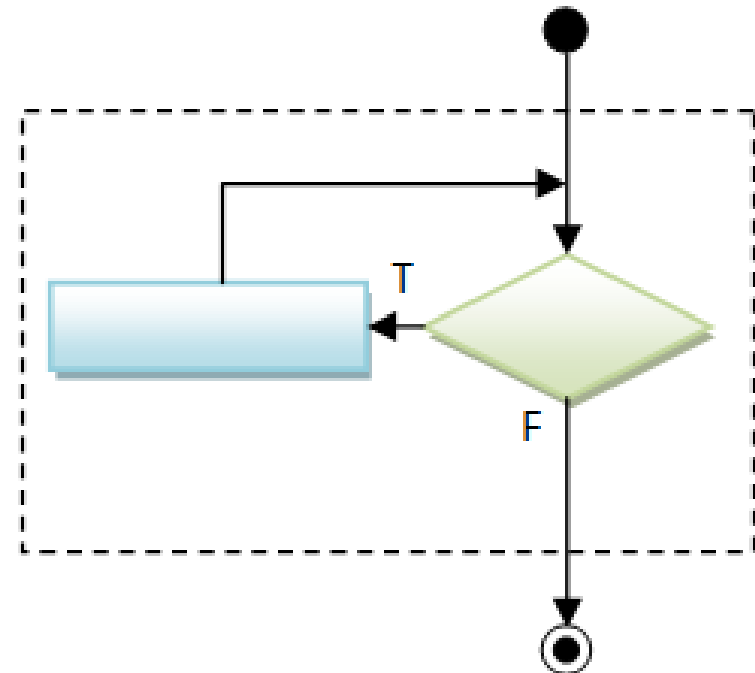
**Sequential**

**Today**



**Conditional (Decision)**

**Learn yourself later (See Appendix)**



**Loop (Iteration)**

# Sequential

- Example: Sum of Squares

$$\sum_{t=1}^3 t^2$$

```
# sum of squares  
t <- 1:3  
y <- sum(t^2)  
print(y)
```

# Sequential

- Example: Sum of Squares

$$\sum_{t=1}^3 t^2$$

```
# sum of squares  
t <- 1:3  
y <- sum(t^2)  
print(y)
```

t	1	2	3
---	---	---	---

# Sequential

- Example: Sum of Squares

$$\sum_{t=1}^3 t^2$$

```
# sum of squares  
t <- 1:3  
y <- sum(t^2)  
print(y)
```

t	1	2	3
t^2	1	4	9
sum(t^2)	14		

# Programming Structure: Functions

- What's a function
  - a logical block of code
  - input -> output
- Why write functions
  - Reusability
  - Abstraction
  - Maintainability
- Example:  $\sum_{t=1}^n t^2$

```
# sum of squares from 1 to n
ss <- function(n) {
  t <- 1:n
  sum(t^2)
}

# calling the ss() function
print(ss(2))
print(ss(3))
```

# Programming Structure: Functions

- What's a function
  - a logical block of code
  - input -> output
- Why write functions
  - Reusability
  - Abstraction
  - Maintainability
- Example:  $\sum_{t=1}^n t^2$

```
# sum of squares from 1 to n
ss <- function(n) {
  t <- 1:n
  sum(t^2)
}

# calling the ss() function
print(ss(2))
print(ss(3))
```

# Programming Structure: Functions

- What's a function
  - a logical block of code
  - input -> output
- Why write functions
  - Reusability
  - Abstraction
  - Maintainability
- Example:  $\sum_{t=1}^n t^2$

```
# sum of squares from 1 to n
ss <- function(n) {
  t <- 1:n
  sum(t^2) # return(sum(t^2))
}

# calling the ss() function
print(ss(2))
print(ss(3))
```



# Turn Ideas into Code

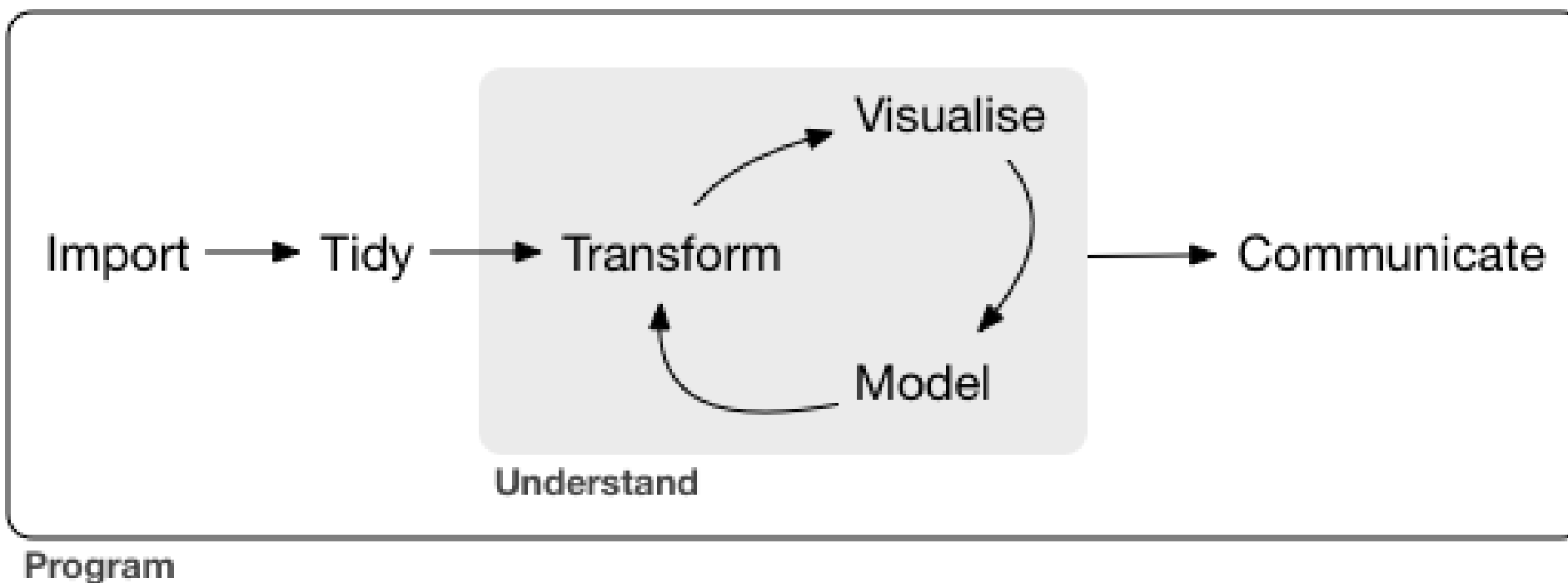
- Solve problems using code: three main ingredients
  - Data & Programming Structure + **Algorithm** (sorting, searching, optimization, etc.)
- Examples
  - Sort a list of integers
  - Generate and solve Sudoku puzzles
  - Implement and backtest a trading rule/algorithm
- For us, in most cases, we solve problems by
  - using other people's algorithm implementations (functions from R packages)
  - Combine algorithms (and data & programming structures) to achieve our goal (still not easy; need practices to write good code.)

# Plan for Today

- Intro to Intro
- How to learn R and a quick walk-through
  - Basics of R programming
  - Data science with R
    - A Typical data analysis workflow
    - Choice of R packages
    - An example: regression analysis
- Learning Road Map and Resources

# Data Science/Analysis Workflow

- Use this workflow to organize your thoughts and code



# An Example: Housing Price & Clean Air

Obs: 506

- Manipulate data
    - Load data
    - Create new columns
    - Filter columns and rows
  - Build models
    - Multiple linear regressions
  - Report and graph
    - Build a publication-ready table for regression results
- |                   |                                      |
|-------------------|--------------------------------------|
| 1. <b>price</b>   | median housing price, \$             |
| 2. <b>crime</b>   | crimes committed per capita          |
| 3. <b>nox</b>     | nitrous oxide, parts per 100 mill.   |
| 4. <b>rooms</b>   | avg number of rooms per house        |
| 5. <b>dist</b>    | weighted dist. to 5 employ centers   |
| 6. <b>radial</b>  | accessibility index to radial hghwys |
| 7. <b>proptax</b> | property tax per \$1000              |
| 8. <b>stratio</b> | average student-teacher ratio        |
| 9. <b>lowstat</b> | % of people 'lower status'           |

# R Packages: Many choices, which one to use

- Often, a task can be achieved using functions in different libraries
  - R is open and extensible!
- Example: load a csv file to a data frame
  - Use [read.csv\(\)](#) function from the `utils` library in Base R
  - Use [read\\_csv\(\)](#) function from the [readr](#) library
  - Use [fread\(\)](#) function from the [data.table](#) library
  - Use [vroom\(\)](#) from the [vroom](#) library

# R Packages: Many choices, which one to use

- Start with the one most people use
- Choose one that is well maintained
  - check document, github, etc. for last update date
  - packages maintained by companies (e.g., RStudio Co.) or academic teams
- Choose one that suits your task

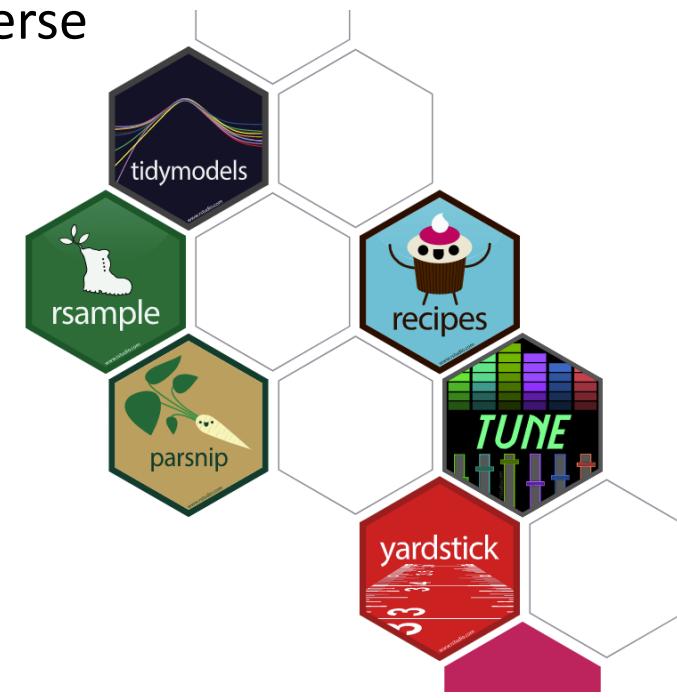
# Great Choice for Data Science Work

- Tidyverse

- “an opinionated collection of R packages designed for data science”
- “All packages share an underlying design philosophy, grammar, and data structures.”
- Handle data manipulation, visualization, and much more
- an eco-system: many package developers started to follow tidyverse principles too

- Tidymodels

- “a collection of packages for modeling and machine learning using tidyverse principles”
- Manage modeling process but does not do modeling itself



# Our Choice: the Regression Example

- Manipulate data ([tidyverse](#) eco-system)
  - Load data ([read\\_csv\(\)](#) from the [readr](#))
  - Create new columns ([mutate\(\)](#) from [dplyr](#))
  - Filter columns and rows ([select\(\)](#) and [filter\(\)](#) from [dplyr](#))
- Build models
  - Multiple regression ([lm\(\)](#) from stats library in R base)
- Report and graph
  - Build a publication-ready table ([huxreg\(\)](#) from [huxtable](#) library)



# Using R packages/libraries

- Install an R library (only need to install a library once)

```
install.packages("Library_name")
```

- Load an R library (before you use a library)

```
library(Library_name)
```

- [CRAN](#) (The Comprehensive R Archive Network)
  - [CRAN Task Views](#)

# Load a CSV file

- [read\\_csv\(\)](#) from the [readr](#)

```
read_csv(file)
```

```
e.g. hprice <- read_csv("hprice.csv")
```

- More about [read\\_csv\(\)](#)
- More about [readr](#)

# Data Manipulation: dplyr basics

- Filter observations (rows): filter()

```
filter(my_dataframe, condition1, ...)
```

```
e.g., hprice_reg <- filter(hprice, price > 20000)
```

- Select variables (columns): select()

```
mutate(my_dataframe, new_var1 = expression1, ...)
```

```
e.g., hprice_reg <- mutate(hprice_reg, lprice = log(price))
```

- Create new variables: mutate()

```
select(my_dataframe, var1, ...)
```

```
e.g., hprice_reg <- select(hprice_reg, lprice, rooms)
```

## Data Manipulation: Data Pipe (%>%)

```
hprice_reg <- filter(hprice, price > 20000)
hprice_reg <- mutate(hprice_reg, lprice = log(price))
hprice_reg <- select(hprice_reg, lprice, rooms)
```



```
hprice_reg <- hprice %>%
  filter(price > 20000) %>%
  mutate(lprice = log(price)) %>%
  select(lprice, rooms)
```

# Regression

- Multiple regressions: [lm\(\)](#) from stats library in base R

```
my_model <- lm(y ~ x1 + x2, data)
```

$$Y_i = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon_i$$

```
my_model <- lm(y ~ x1 + x2 + I(x1 * x2), data)
```

$$Y_i = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + \epsilon_i$$

- Regression result summary: `summary()`

Ref. <https://faculty.chicagobooth.edu/richard.hahn/teaching/FormulaNotation.pdf>

# Report

- Summary table
  - [Summary for lm\(\)](#): `summary(my_model)`
- publication-ready table: [huxreg\(\)](#) from [huxtable](#) library

```
huxtable(my_model1, my_model2, ...)
```

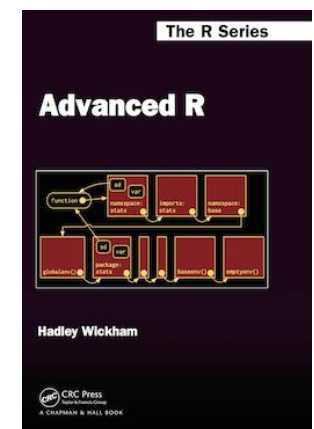
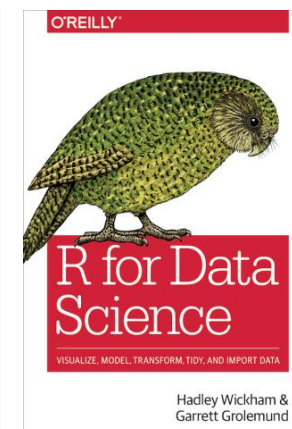
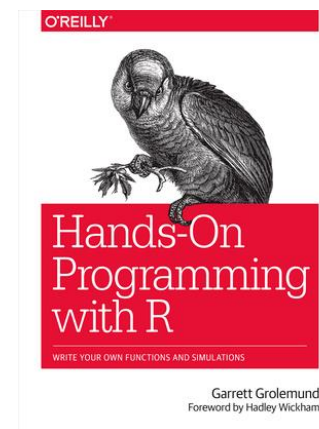
# Plan for Today

- Intro to Intro
- How to learn R and a quick walk-through
  - Basics of R programming
  - Data science with R
- Learning Road Map and Resources

# Learning Road Map (Three Free Books)

- Step 1. Basic R programming skills (Never programmed before? Start here.)
  - Data and programming structure; how to turn an idea into code;
  - Book: [Hands-On Programming with R](#)
- Step 2. R Data Science skills
  - Data wrangling, modeling, and visualization/reporting; Best practice;
  - Book: [R for Data Science](#)
- Step 3. Take your R Skill to the next level
  - Book: [Advanced R](#)

Other free books check [bookdown.org](http://bookdown.org) often





# Free Learning Resource

- [RStudio Education](#)
  - [Choose Your Learning Paths](#)
- [RStudio Video Resources Site](#)
- More free R books? Check [bookdown.org](http://bookdown.org) often
- Coursera: Search R and learn
  - free for [UofT students](#) (mostly always free if you just audit the courses)
- Twitter (a few seeds: [#rstat](#), [@hadleywickham](#), [@WeAreRLadies](#))

# Appendix

- Programming Structure Continued
  - Conditional
  - Iteration

# Conditional (if...else...)

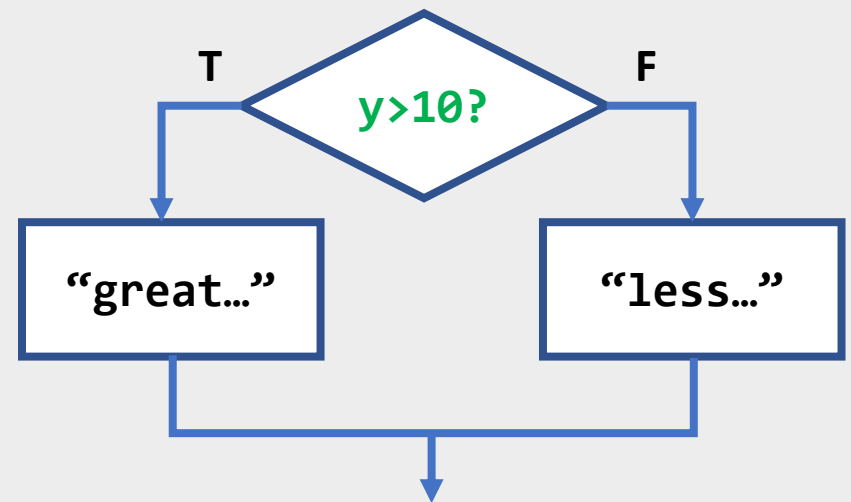
```
if (cond) {  
    # run here if cond is TRUE  
} else {  
    # run here if cond is FALSE  
}
```

```
# y greater than 10?  
if (y > 10) {  
    print("greater than 10")  
} else {  
    print("less or equal to 10")  
}
```

# Conditional (if...else...)

```
if (cond) {  
    # run here if cond is TRUE  
} else {  
    # run here if cond is FALSE  
}
```

```
# y greater than 10?  
if (y > 10) {  
    print("greater than 10")  
} else {  
    print("less or equal to 10")  
}
```



# Conditional (if...else if...else...)

```
if (cond1) {  
    # run here if cond1 is TRUE  
} else if (cond2) {  
    # run here if cond1 is FALSE but cond2 is TRUE  
} else {  
    # run here if neither cond1 nor cond2 is TRUE  
}
```

# Iteration

```
for (var in seq) {  
  do something  
}
```

```
while (cond) {  
  do something if cond is TRUE  
}
```

```
# sum of squares  
t <- 1:3  
y <- 0  
  
for (x in t) {  
  y <- y + x^2  
}  
  
print(y)
```